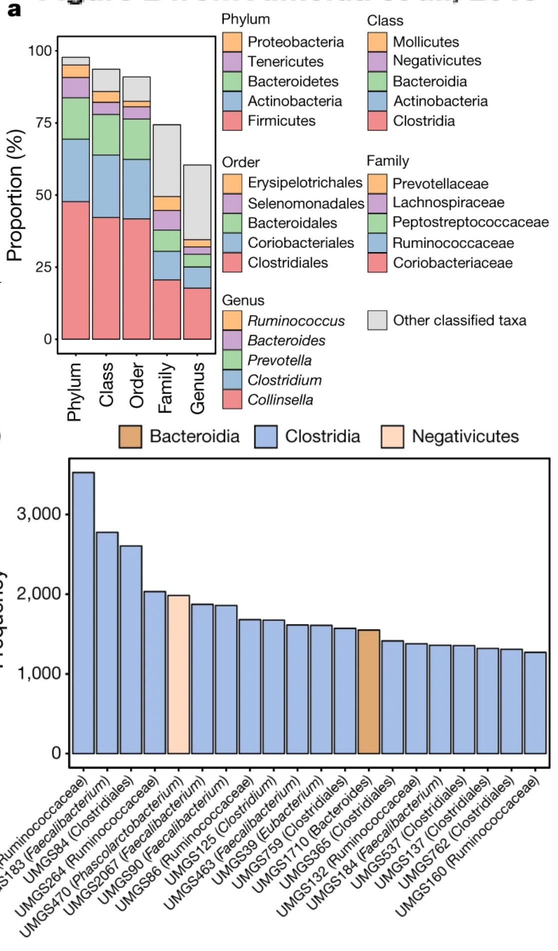
BIOIN 401 Final Report
Bryland Schoneck
1473495
April 27, 2021

Introduction:

For this class my team and I decided to replicate: *A new genomic blueprint of the human gut microbiota* from Nature[1], which will be known as "the paper". As a type 1 diabetic, I am always reading and researching new methods and implications of helping to prevent, control or cure diabetes. One topic of research that intrigued me most was that of how our gut microbiome affects new and long term diabetics. Though the paper did not discuss medical applications of their discoveries, it allows me to better understand the diversity and function of our gut bacteria, which I can apply to my future research. From a more computational perspective, what interested me was the bioinformatic software that the paper used, in particular the nuances that allowed for greater speed and efficiency. I have always been obsessed with efficiency, always going the extra mile to make my programs or pipelines just a little bit faster, so finding out how these softwares accomplished this was intriguing. For the purposes of this course only part of the paper was replicated using intermediary data provided by the paper. Figures 2 and 5, which focused on taxonomic analysis/species prevalence and functional characterization respectively, were our targets as they provided the most interesting and useful information. The paper starts with assembling and binning from the reads of thousands of metagenomic datasets from a diverse background. After quality assessment, filtering out assemblies that were already known and dereplication they were left with 1952 unclassified metagenomic species (UMGS). To obtain the predicted genes and proteins used in further analysis the paper used Prodigal[2] The taxonomic analysis (Fig.2a) relied on both marker genes and protein-level matches using both CheckM[3] and

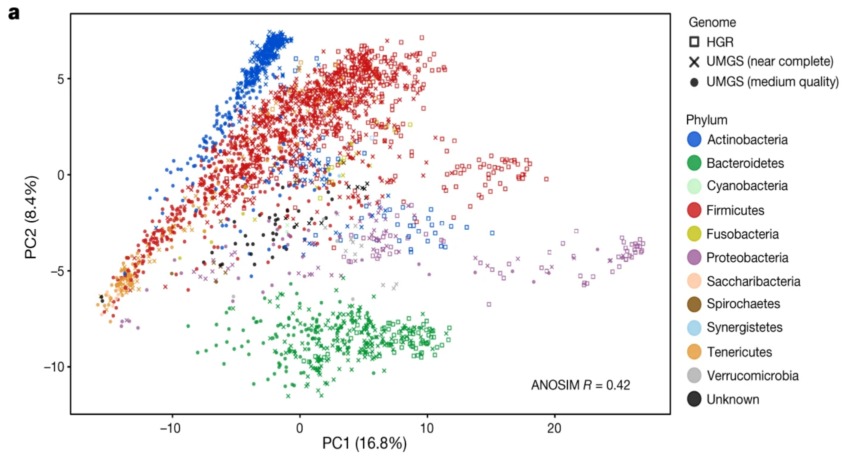
Figure 2 from Almeida et al., 2019

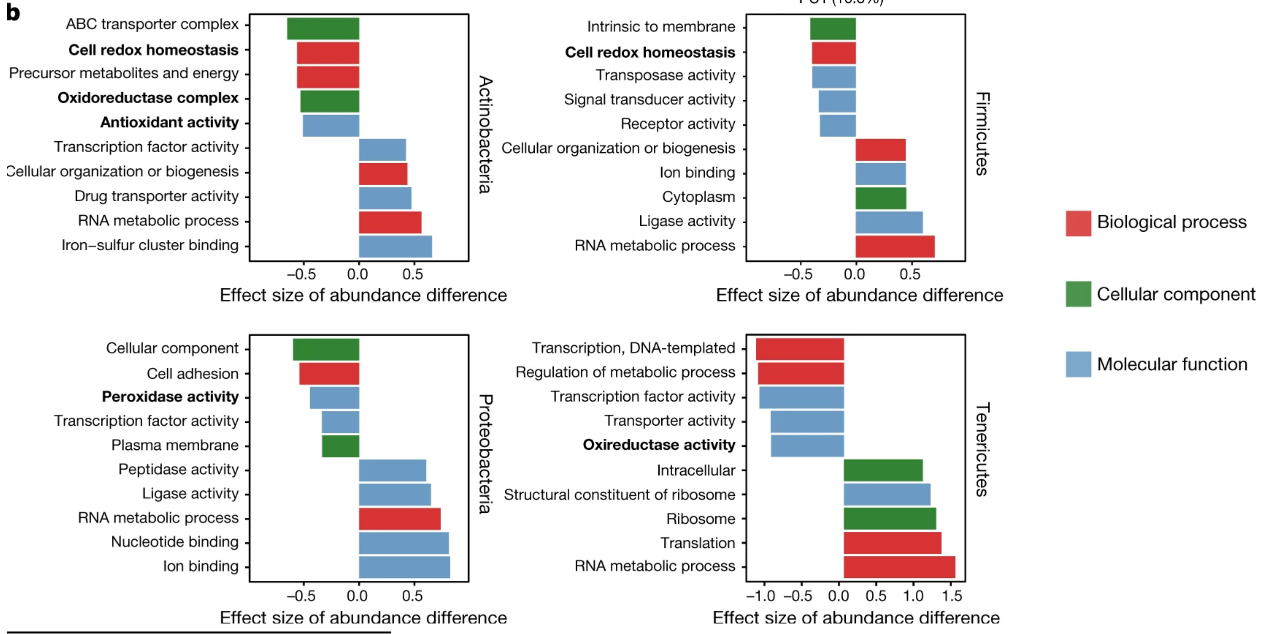


[1] Almeida, A., Mitchell, A.L., Boland, M. et al. A new genomic blueprint of the human gut microbiota. Nature 568, 499–504 (2019).
[2] Hyatt, D. et al. Prodigal: prokaryotic gene recognition and translation initiation site identification. BMC Bioinformatics 11, 119 (2010).
[3] Parks, D. H., Imelfort, M., Skennerton, C. T., Hugenholtz, P. & Tyson, G. W. CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes. Genome Res. 25, 1043–1055 (2015).

DIAMOND[4] softwares respectively. CheckM uses lineage specific marker genes to place our genomes within its internal reference tree. As complimentary data DIAMOND through UniProtKB[5] uses blastp to align proteins to our predicted proteins. Manual inspection/comparison of the results were used to find the UMGS' most likely taxonomic lineage. In cases of inconsistencies between CheckM and DIAMOND, CheckM was taken over DIAMOND. Species prevalence (Fig. 2b) made use of BWA MEM[6] along with samtools[7] was used to map the raw reads of the metagenomic datasets to the UMGS and statistically calculate whether the UMGS was present or absent from the metagenomic data. Functional analysis (Fig.5) was performed by InterProScan[8] to annotate Gene Ontology (GO) terms for the UMGS and human gut reference (HGR) genomes. These annotations were translated into Genome Properties (GP). Principal component analysis was used to produce Fig. 5a based on the GP

Figure 5 from Almeida et al., 2019





[4] Buchfink, B., Xie, C. & Huson, D. H. Fast and sensitive protein alignment using DIAMOND. Nat. Methods 12, 59–60 (2015).
[5] The UniProt Consortium. UniProt: the universal protein knowledgebase. Nucleic Acids Res. 45, D158–D169 (2017).
[6] Li, H. & Durbin, R. Fast and accurate long-read alignment with Burrows–Wheeler transform. Bioinformatics 26, 589–595 (2010).
[7] Li, H. et al. The Sequence Alignment/Map format and SAMtools. Bioinformatics 25, 2078–2079 (2009).
[8] Jones, P. et al. InterProScan 5: genome-scale protein function classification. Bioinformatics 30, 1236–1240 (2014).

distribution of UMGS and HGR. Fig. 5b shows the differential abundance of the GO terms between the UMGS and HGR genomes.

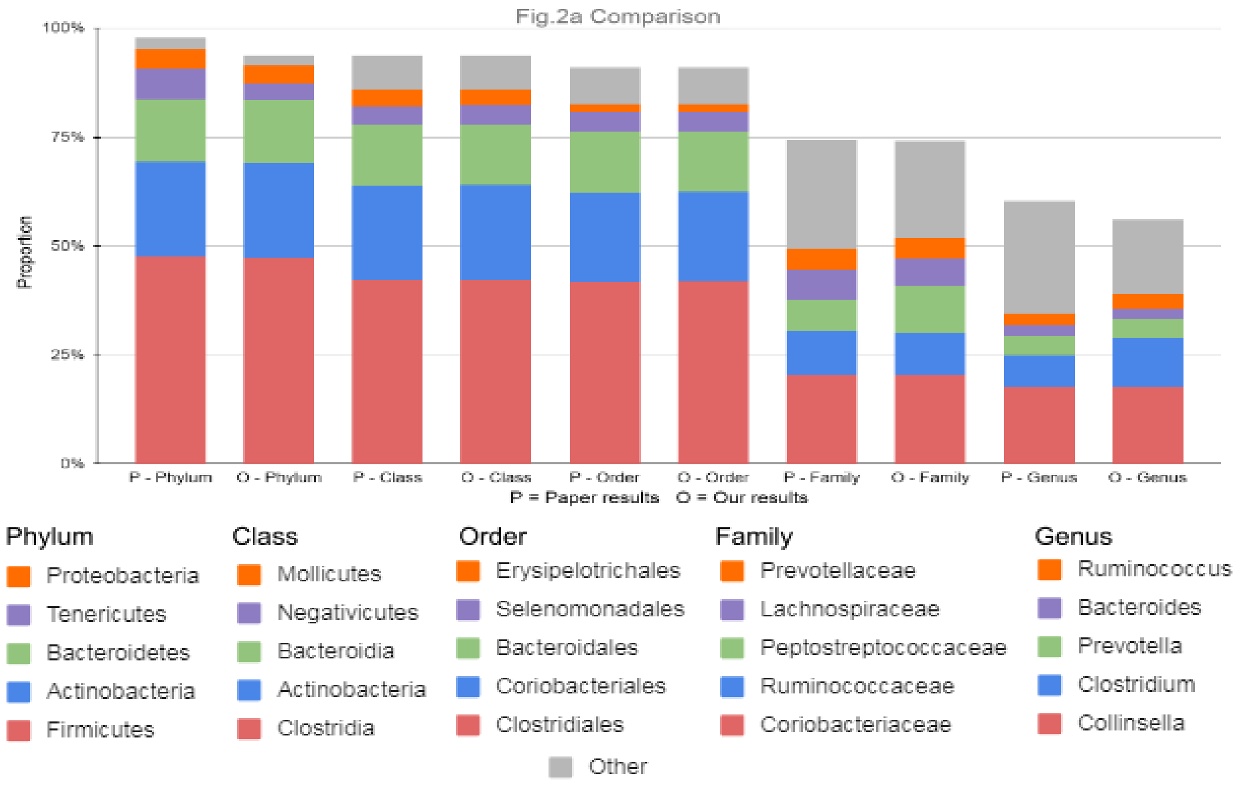
Contributions:

For this report we worked in a team of two that consisted of Harrison Peters and myself, Bryland Schoneck. In general we each worked on one figure each, with Harrison working on figure 5 and figure 2 being worked on myself. Initially due to my computing power being obsolete compared to Harrison`s we had planned for Harrison to be running the computation while I worked on parsing and analysing the raw output data. This plan shifted when the manual curation of figure 2 took much longer than expected and we decided that I would continue with figure 2 and Harrison would move on to figure 5. We both did research into the methods of which the authors performed their assemblies and dereplication. Harrison also decided to compute extended data figure 8 while InterProScan ran in the background. While most of our work was done independently, we consistently updated each other on our progress as well as helped each other with computational and paper related issues.

Results and Methods:

Figure 1 shows the comparison of the paper's results from Fig.2a and our results from attempting to replicate their data. The exact counts of each taxa group can be found in supplementary data Table 1. The chart shows almost exact counts for Phylum, Class and Order columns whereas Family and Genus start to show greater differences as expected as the taxonomic lineage gets more specific and small differences matter more. The initial data we used was intermediary data taken from the paper's ftp server. The first step of this process uses CheckM's tree and tree_qa functions. The tree function places our 1,952 UMGS on CheckM's internal reference phylogenetic tree based on lineage specific markers, after which tree_qa

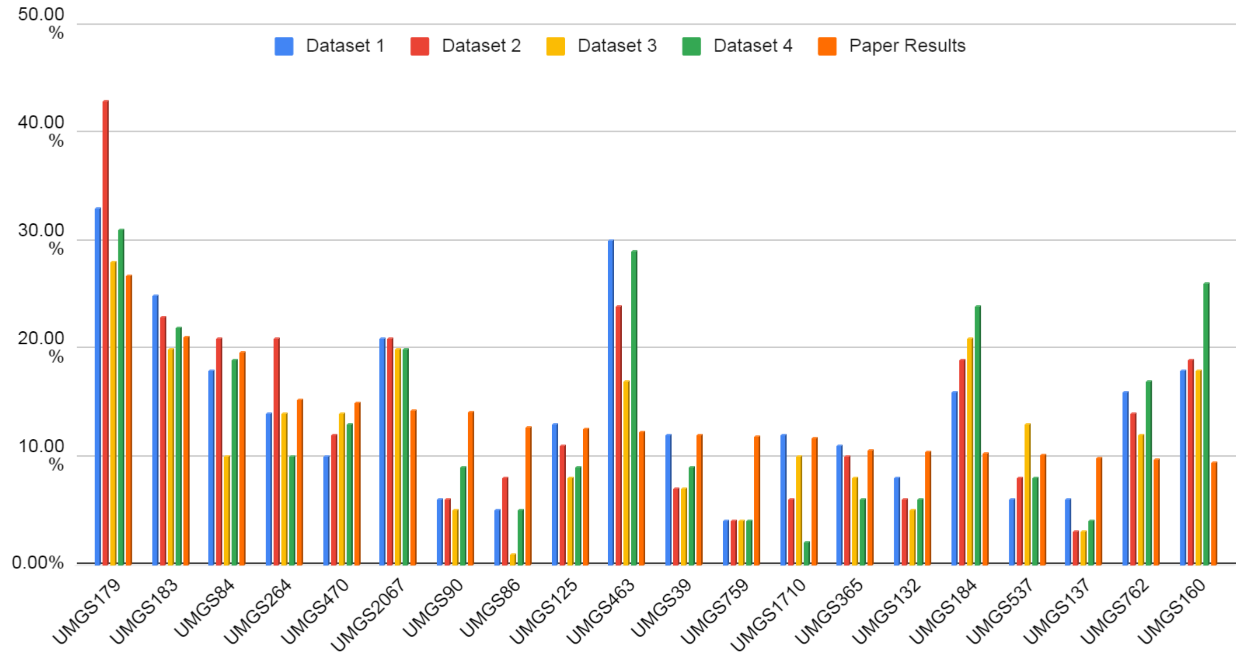
Fig.1



outputs the number of marker genes used for tree placement as well as it's taxonomic lineage. For this step an assumption was made that the paper used CheckM's full reference tree, which requires 40Gb of RAM, which we did not have access to. To compromise under our constraints we opted to use the reduced tree, which would only require 16Gb of RAM. There is little to no documentation about how reduced the tree is compared to the original or what exactly has been stripped from the tree. The authors did not publish their intermediary results from only CheckM so we were unable to know for sure the exact impact this step had on the inconsistencies of our results. The second step used the software DIAMOND along with extensive manual curation to supplement the taxonomic analysis from the previous step. This step makes use of DIAMOND's makedb and blastp functions which made a local database and queried it finding protein matches to our predicted proteins. The database we used was UniProtKB (release 2018_04), the same as the paper. DIAMOND was only done to those UMGS that were classified at least at the Class rank. Additionally to this subsection, to save time in computation we only used those UMGS that did not already have a perfect match with CheckM alone, as the result of DIAMOND would not override CheckM's. A genus level threshold was applied to the results of the blastp: more than 50% of proteins having an e value < 1x10^-5, a seq id > 40% and a query coverage > 50%. The paper originally started with 2,068 MGS in which they used a species level threshold to determine if any of their MGS were already classified. They found that our 1,952 UMGS did not match any known species. We did not include this step into our analysis because the authors only published intermediary data of the 1,952 UMGS. The blastp results gave us only a UniProtKB id, which we had to again query the UniProtKB for a taxonomic identity. After trying to query that database for the taxonomic ids, the time needed to search well over a million ids was completely infeasible. The compromise we had to make was to search these ids on UniProtKB's current release using many manual batch queries. This change in release undoubtedly resulted in more inaccuracies of our results as some of the ids were either missing or obsolete. The method of determining the most likely taxonomic lineage from thousands of matched genera was very poorly documented in the paper. Research on the topic of how this is generally accomplished was extremely varied in which many different options were reported to be used. Attempting some of which on my own like: only using the top result or top 10 results left or using results within a threshold of the top result. All of these options had a common problem in that much of the data was seemingly arbitrarily thrown out or not considered and came from people finding an already identified species. We eventually committed to an idea that most closely resembled what the paper was striving for, where all results were considered but only taxonomic lineages with matches in over 50% of our predicted proteins. What caused me so much confusion from the beginning was the idea that the result needed to point at one specific lineage for the answer, but this method gave a list of possible answers. Using manual curation, the lineage that most closely resembled that of CheckM's result was used to fill in the missing details from CheckM. After combining the results of both CheckM and DIAMOND into our final result and comparing it to the results from the paper the inaccuracies from the compromises we needed to make were more apparent. Our results showed that many of the infrequent genera of the UMGS that were not able to be identified from either source, but was present in the paper's result likely points to the reduced tree of CheckM missing those that were not as common. Another noticeable discrepancy is the relatively larger overabundance of the genus Clostridium, which can be reasonably attributed to the genus being extremely diverse and complex as the exact nature to which many of these species, even up to the order rank, should be classified has been in debate and has undergone constant rearrangement. According to another paper[9] published only 2 months after Almeida, A. et al.'s, the genus Clostridium should be split further, major differences in pathogens classified as the same species, many inconsistencies in the evolution and taxonomy, and identified significant differences within other genera that were once Clostridium.

Figure 2 shows the species prevalence of the top 20 most prevalent UMGS within the original metagenomic datasets they were assembled from. Four random subsamples of 100 metagenomic datasets are compared to the paper's result from all 13,133 metagenomic datasets. Statistical calculation based on the very small sample size reports with 95% confidence a margin of error of 10%. Most results from our analysis fall within the margin of error and almost all when taken the average of the four subsamples. Exact counts and percentages can be found in supplementary data Table 2a and 2b respectively. Starting data for this analysis consists both of

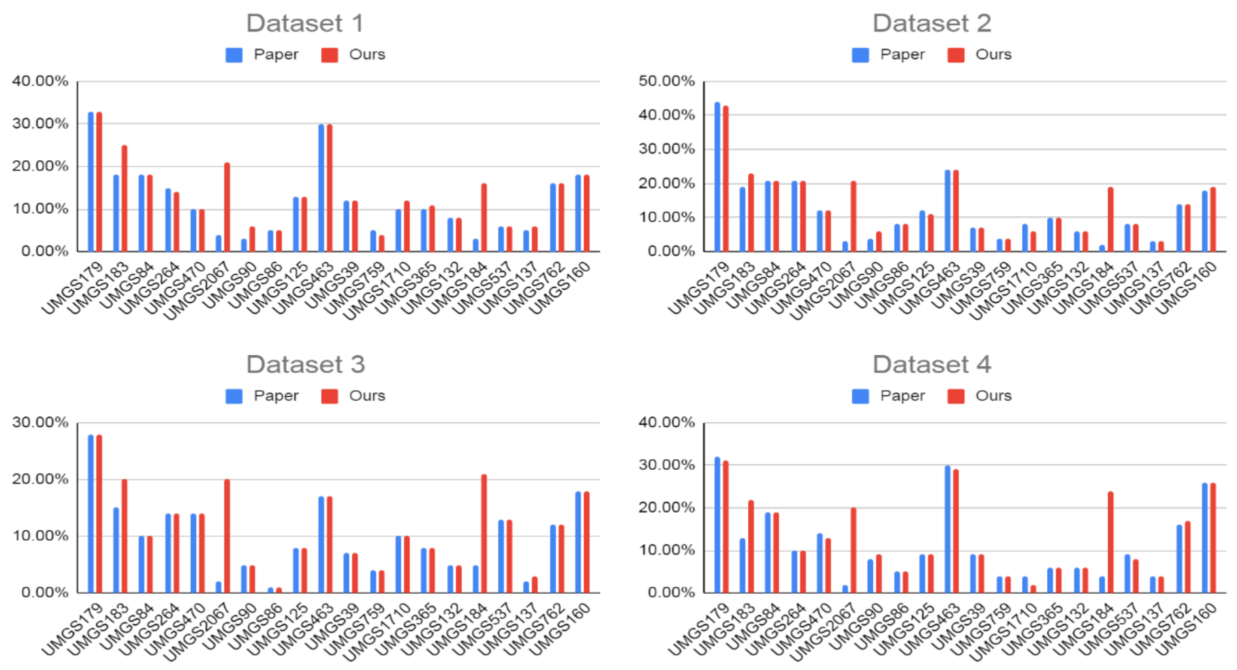
Fig. 2



intermediary data supplied by the authors as well as forwards and reverse raw reads from stool samples from many studies accessed via the European Nucleotide Archive (ENA). The main

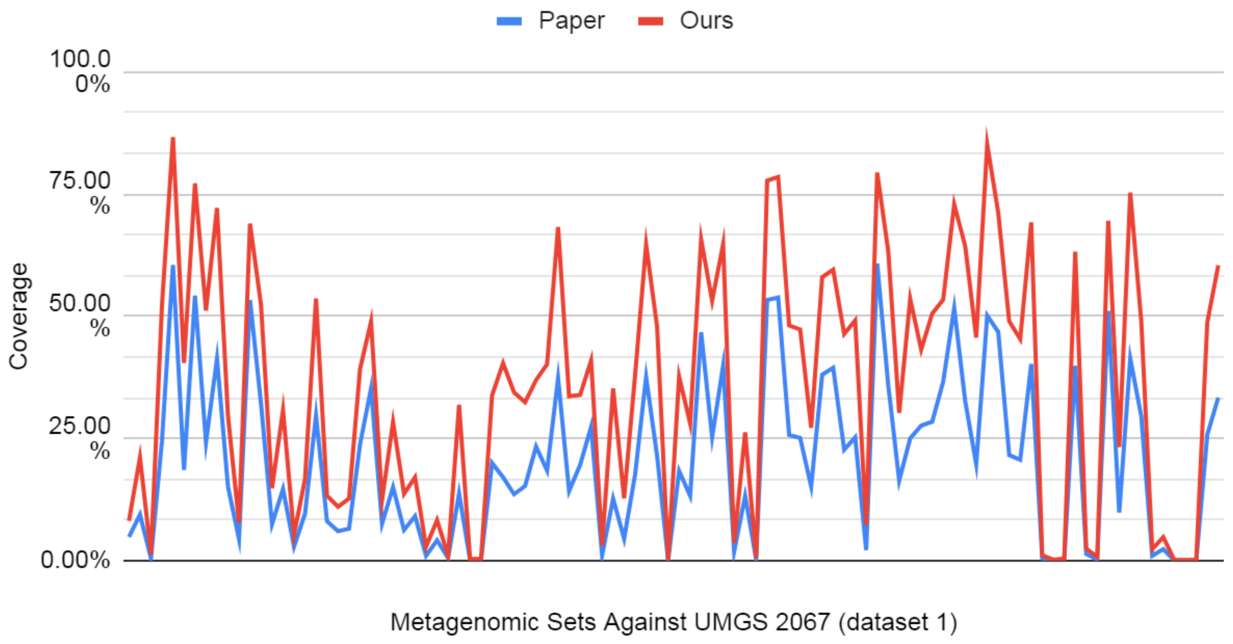
[9] Cruz-Morales, Pablo, et al. "Revisiting the Evolution and Taxonomy of Clostridia, a Phylogenomic Update." Genome Biology and Evolution, edited by Eric Bapteste, vol. 11, no. 7, 2019, pp. 2035–44. Crossref, doi:10.1093/gbe/evz096.

strategy behind this method is mapping the raw reads back to our UMGS catalogue and determining presence/absence based on genome coverage, mean read depth and depth coefficient of variation, which is the standard deviation of read depth divided by the mean. BWA MEM was the backbone of the mapping and statistical analysis pipeline in which it used a method of seeding alignment and finding maximal exact matches and extending them using a local sequence alignment algorithm. BWA MEM requires a reference db of the UMGS and the authors were not clear on the exact nature or format to which this reference database should be indexed and assembled. Initially, we started with a reference database only containing the top 20 most prevalent UMGS to save on time and computation as my resources at the time could not handle much more. This decision had the assumption that we did not need the rest of the UMGS because we only cared about the top 20. This assumption was found to be false as leaving out the rest of the UMGS in the reference database resulted in great uniform increase in coverage in all metagenomes and UMGS leading to very skewed results. Upgrading to better hardware, I was able to rerun the data with the full catalogue of UMGS in the reference database. Samtools was then used to format and extract our statistical values. From those statistical values, genome coverage, mean read depth and depth coefficient of variation, penalty scores for depth and variation were calculated from (100% − genome coverage) multiplied by the log(mean depth) or (100% − genome coverage) multiplied by the depth coefficient of variation, respectively. To determine the precedence of a UMGS, boundary thresholds were set: genome coverage < 60%, depth and variation penalty scores in the 99th percentile. Due to the lack of much intermediary data provided by the paper I was unable to fully compare my results during the process of the pipeline to pinpoint exact locations a discrepancy might have occurred. From the intermediary data available to me and rewriting parts of their non-functional R scripts, I was able to get binary results for each metagenomic dataset whether or not each UMGS was present. Figure 3 shows the comparison of the prevalence of the top 20 UMGS for our results to the papers. For all but



two UMGS the numbers are extremely comparable. Although I initially expected I should get exact agreement, the numbers that are slightly off are easily explained by the very small sample size I used. Due to part of the constraint thresholds for prevalence are based on other UMGS-metagenomic pairs as they need to be in the 99th percentile of two categories the absence of some UMGS is to be expected. Furthermore, the UMGS were in exact agreement with which metagenomic sets they were a part of. This explained almost everything except for UMGS 2067 and UMGS 184 which both have extreme discrepancies between them. Looking into the statistical data output from the pipeline, a pattern of uniform increase in all categories but mainly coverage. Although this discrepancy in coverage was causing UMGS being falsely identified in our results, it was very clear that in all cases in which a UMGS was falsely identified, there was also a significant spike in coverage in the paper's results, often just under the threshold coverage. Figure 4 shows an example of this by comparing UMGS 2067's coverage from the paper's and my results for dataset 1. This was evidence that I was close but why did only two of my twenty UMGS have this issue? Further research down every rabbit hole I could find led me to many issue reports on the BWA github claiming problems with reproducibility. Mainly these issues revolved around the idea that BWA MEM uses random seeding along with randomly choosing between alignments of the same score. The first issue is solvable as an argument could be used to set your own specific seeding size, so your results become more deterministic and reproducible, but the problem remained as the authors did not use this setting. The issues were apparently amplified with genomes that have highly repetitive sequences. Since a lot of this is not within the realm of my capabilities to solve for certain along with lacking intermediary results and no real indication of how the authors exactly went about with their analysis, my final guess is that the UMGSs 2067 and 184 must contain highly repetitive regions that caused these inconsistencies.

Figure 4



An important part of the paper that we did not have the opportunity to do a complete analysis of was the process in which the authors assembled, binned and dereplicated the metagenomes to get almost 2000 new before discovered species. Starting with the assembly with SPAdes[10,11] there are two main ways in which the authors could go about completing this. The first and better way would be to compile all raw reads from the metagenomic datasets and assemble from all available data. This method allows you to possibly get more and better quality assemblies but at the expense of a lot of time and money. The other way makes assemblies for each dataset individually. This method is much faster and much cheaper than the previous but has the chance to throw out a lot of more and potentially higher quality assemblies. The choice between the two methods comes down to how worth those potential genomes are, relative to the amount of time needed to get them. To figure out which method the authors used, we found that each UMGS had an accession and bin number as an alternative name, to which we could trace back to find exactly which metagenome it came from. This shows that the authors decided to go with method two, which may seem lazy as they are some of the most well funded researchers but the time investment to combine and assemble over 13000 datasets may have been infeasible to compute. The binning and quality assessment was pretty straightforward, but while I researched some of these softwares for literature week, I noticed a distinct lack of important and useful information on how exactly the software works or what kind of data ranges the specifically look for or calculate. CheckM was used to quality check their bins, and gave the authors two groups: 40,029 near-complete metagenome-assembled genomes (MAGs) and 52,347 medium quality MAGs. Once the authors had their MAGs, they attempted to assign each to a known genomes in a human-specific reference (HR) database and RefSeq. After, they found that 11,888 of the near-complete MAGs were not assigned. Dereplication of these MAGs by a process of grouping MAGs that seemed like they would be the same species and chose the highest quality one to represent that species, after which they had 1,175 MGS. This is also a disappointing revelation because they seemingly threw away 90% of their data, especially because they already took possibly lower quality assemblies to begin with. They then supplement these numbers by putting their medium quality MAGs through the same process, adding 893 more MGS. It is stated that they do dereplication on each group of MAGs separately, which does not seem consistent with having only a singular representative MGS for each species. This makes little sense to me as they assembled these genomes separately, meaning that there is more room for both near-complete and medium quality MAGs to represent the same species but then dereplicate the groups separately if your goal was one of each species.

[10] Bankevich, A. et al. SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. J. Comput. Biol. 19, 455–477 (2012).
[11] Nurk, S., Meleshko, D., Korobeynikov, A. & Pevzner, P. A. metaSPAdes: a new versatile metagenomic assembler. Genome Res. 27, 824–834 (2017).

Closing:

Bioinformatics is hard. I knew that before enrolling in this class, but now I can say I have an appreciation for the kind of work these people do. Although, it might seem like the authors of this paper have frustrated me, which they have on multiple occasions, I can understand the need to compromise and cut corners for your results and not even the most well funded labs are immune to this. It also shows me that even an overworked, unpaid undergraduate can make progress in these advanced and complex topics. It was a stark contrast coming into the world of bioinformatics from computer science since I assumed they would be so similar. I am a little disappointed at the level of ego, secrecy and competition these software companies operate in, whereas a lot of what I find in computer science is open sourced, and community advancement towards a common goal instead of the race to find out first. I don't know how realistic that kind of future is for bioinformatics but I hope it gets there someday.

NOTE: All figures in the main report are also in the supplementary blown up to see easier.